

Laboratorio Nro. 2

Notación O grande



Objetivos: 1. Usar ecuaciones de recurrencia para determinar la complejidad en tiempo y espacio de algoritmos definidos de forma recursiva. 2. Usar la notación O para encontrar formalmente la complejidad asintótica en espacio y memoria de algoritmos. 3. Realizar estudios empíricos para validar una hipótesis sobre el comportamiento en tiempo de ejecución de un algoritmo con varios tamaños de problema.



Consideraciones: Lean y verifiquen las consideraciones de entrega,



Leer la Guía



Trabajo en
Parejas



Si tienen reclamos,
regístenlos en
<http://bit.ly/2q4TTKf>



Ver
calificaciones
en Eafit
Interactiva



En el GitHub
docente,
encontrarán la
traducción de
los Ejercicios
en Línea

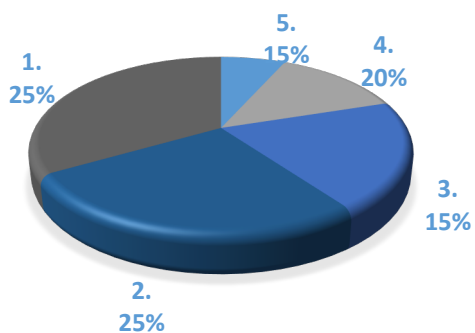


Hoy, plazo de
entrega



Subir el **informe pdf** en la carpeta **informe**, el **código del ejercicio 1** en la carpeta **codigo** y el **código del 2** en la carpeta **ejercicioEnLinea**

Porcentajes y criterios de evaluación



- 1. Simulacro sustentación proyecto
- 2. Análisis de complejidad
- 3. Código de laboratorio
- 4. Simulacro de parcial
- 5. Ejercicio con juez en línea

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

1. Simulacro de Proyecto

Códigos para entregar en GitHub en la carpeta `codigo`

1 Simulacro de Proyecto Códigos para entregar en GitHub en la carpeta `codigo`



En la vida real, la documentación de software hace parte de muchos estándares de calidad como CMMI e ISO/IEC 9126



Vean Guía numeral 3.4



Código del ejercicio en línea en **GitHub**. Vean Guía en numeral 4.24



Documentación opcional. Si lo hacen, utilicen **Javadoc** o equivalente. No suban el HTML a GitHub.



No se reciben archivos en **.RAR** ni en **.ZIP**



Utilicen Java, C++ o Python

Una de las principales características del videojuego *Silent Hill 3* es la niebla, que para modelarla se utiliza la transparencia. Para poder renderizar texturas o efectos que usen transparencia, se necesita renderizar los elementos en orden del que está más atrás al que está más adelante para tener una visualización correcta.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 1

Código ST0245

El criterio de ordenamiento de los objetos 3D es una métrica que represente la profundidad. El problema de escoger el orden de los objetos a renderizar se reduce a un problema de ordenamiento de números. Existen diversos algoritmos para ordenamiento de números.

Para poder que la renderización de los objetos 3D sea eficiente, se necesita elegir un algoritmo que sea eficiente para este problema. Dos algoritmos conocidos para este problema son Insertion Sort y Merge Sort. Nuestra tarea es decidir cuál es más eficiente para este problema



Imagen extraída de <https://bit.ly/2ThpLw9>



Extiendan el código existente para medir los tiempos (en milisegundos) de *Insertion Sort* y *Merge sort* con 20 tamaños del problema. Si para todos los tamaños del problema se demora 0 ms, no son adecuados.



Nota:

- Identifiquen valores apropiados para el tamaño del problema
- Tomen los tiempos para 20 tamaños del problema diferentes.
- Hagan una gráfica en Excel para cada algoritmo y pasarla a Word luego
- Entreguen el informe en formato PDF

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

2. Simulacro de Maratón de Programación sin documentación HTML, en GitHub, **en la carpeta `ejercicioEnLinea`**

2 Simulacro de Maratón de Programación sin documentación HTML, en GitHub, en la carpeta ejercicioEnLinea



Vean Guía
numeral 3.3



No se requiere
documentación para
los ejercicios en línea



Utilicen Java,
C++ o Python



No se reciben archivos
en **.PDF ni .TXT**



**Resolver los
problemas de
CodingBat
usando
Recursión**



Código del ejercicio en
línea en **GitHub**. Vean
Guía en numeral 4.24

! **Nota:** Si toman la respuesta de alguna fuente, deben **referenciar** según el **tipo de cita**. Vean Guía en **numerales 4.16 y 4.17**

2.1 Resolver al menos 5 ejercicios del nivel *Array 2* de la página *CodingBat*:
<http://codingbat.com/java/Array-2>

2.2 Resuelvan mínimo 5 ejercicios del nivel *Array 3* de la página *CodingBat*:
<http://codingbat.com/java/Array-3>

! **Nota:** La complejidad máxima para ejercicios de *Array 3* es $O(n^2)$

3. Simulacro de preguntas de sustentación de Proyecto en la carpeta informe

3 Simulacro de preguntas de sustentación de Proyecto en la carpeta informe



Vean **Guía**
numeral 3.4



Exporten y entreguen
informe de laboratorio en
PDF, en español o Inglés



Si hacen el **informe en español**, usen la **plantilla en español**



No apliquen **Normas Icontec** para esto

Si hacen **el informe en inglés**, usen a **plantilla en inglés**

Sobre el Simulacro de Proyectos

- 3.1** De acuerdo a lo realizado en el **numeral 1**, construya una tabla donde muestre, para cada algoritmo, cuánto se demora para cada uno de los 20 valores del tamaño del problema.



En la vida real, las gráficas son una forma simple y concisa de representar resultados cuantitativos en Ingeniería de Sistemas

- 3.2** Grafiquen los tiempos que tomó en cada algoritmo para los diferentes tamaños del problema. Grafiquen el Tamaño de la Entrada Vs. Tiempo de Ejecución.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473



En la vida real, bases de datos relacionales como *MySQL* guardan los datos ordenados, usando como criterio la llave primaria. Como las bases de datos pueden almacenar millones de elementos, es importante entender la eficiencia de los algoritmos de ordenamiento

- 3.3** Teniendo en cuenta lo anterior, ¿Qué tan eficiente es *merge sort* con respecto a *insertion sort* para arreglos grandes?
- 3.4** ¿Es apropiado usar *insertion sort* para un videojuego con millones de elementos?
- 3.5** Para arreglos grandes, ¿en qué caso *insertion sort* es más rápido que *merge sort*? ¿cómo deben ser los datos para que *insertion sort* sea más rápido que *merge sort*?

Sobre el simulacro de maratón de programación


- 3.6** Expliquen con sus propias palabras cómo funciona el ejercicio de Array 3 de Coding Bat llamado *maxSpan* y ¿por qué?


! **Nota:** Recuerden que deben explicar su implementación en el informe PDF

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 1
Código ST0245

3.7  Calculen la complejidad de los Ejercicios en Línea de los numerales 2.1 y agréguenla al informe PDF

3.8  Expliquen con sus palabras las variables (qué es 'n', qué es 'm', etc.) del cálculo de complejidad del ejercicio anterior.



Ejemplos de su respuesta:

“n es el número de elementos del arreglo”,

“V es el número de vértices del grafo”,

“n es el número de filas de la matriz y m el número de columnas”.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas

Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627

Tel: (+57) (4) 261 95 00 Ext. 9473

4. Simulacro de parcial en informe PDF

4 Simulacro de Parcial en el informe PDF



Para este simulacro, agreguen ***sus respuestas*** en el informe PDF.



El día del Parcial no tendrán computador, JAVA o acceso a internet.

- 4.1** Supongamos que $P(n,m)$ es una función cuya complejidad asintótica es $O(n \times m)$ y $H(n,m)$ es otra función cuya complejidad asintótica es $O(m \cdot A(n,m))$, donde $A(n,m)$ es otra función.

¿Cuál de las siguientes funciones, definitivamente, **NO** podría ser la complejidad asintótica de $A(n,m)$ si tenemos en cuenta que $P(n,m) > H(n,m)$ para todo n y para todo m ?

► **Elija la respuesta que considere acertada:**

- a) $O(\log n)$
- b) $O(\sqrt{n})$
- c) $O(n+m)$
- d) $O(1)$

- 4.2** Dayla sabe que la complejidad asintótica de la función $P(n)$ es $O(\sqrt{n})$. Ayúdenle a Dayla a sacar la complejidad asintótica para la función $mystery(n,m)$.

```
void mystery(int n, int m) {
    for(int i = 0; i < m; ++i){
        boolean can = P(n);
        if(can)
            for(int i = 1; i * i <= n; i++)
                //Hacer algo en O(1)
        else
            for(int i = 1; i <= n; i++)
                //Hacer algo en O(1)
    } }
```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

La complejidad de $\text{mystery}(n,m)$ es:

- a) $O(m+n)$
- b) $O(m \times n \times \sqrt{n})$
- c) $O(m+n+\sqrt{n})$
- d) $O(m \times n)$

4.3 [Opc] El siguiente algoritmo imprime todos los valores de una matriz. El tamaño de la matriz está definida por los parámetros largo y ancho.

Teniendo en cuenta que el largo puede ser como máximo 30, mientras que el ancho puede tomar cualquier valor, ¿Cuál es su complejidad asintótica en el peor de los casos del algoritmo?

```
01 public void matrices(int[][] m, int largo, int ancho)
02   for(int i=0; i < largo; i++)
03     for(int j=0; j < ancho; j++)
04       print (m[i][j]);
```



Elija la respuesta que considere acertada:

- a) $O(\text{largo})$
- b) $O(\text{ancho})$
- c) $O(\text{largo} + \text{ancho})$
- d) $O(\text{ancho} \times \text{ancho})$
- e) $O(1)$

4.4 Sabemos que $P(n)$ ejecuta $n^3 + n$ pasos y que $D(n)$ ejecuta $n+7$ pasos. ¿Cuál es la complejidad asintótica, en el peor de los casos, de la función $B(n)$?

```
public int B(int n)
    int getP = P(n);
    int ni = 0;
    for(int i = 0; i < n; ++i)
        if(D(i) > 100){
```

PhD. Mauricio Toro Bermúdez


Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 1
Código ST0245

```

        ni++;
    int nj = getP + D(n) * ni;
    return nj;

```

 **Elija la respuesta que considere acertada:**


- a) $O(n^4)$
- b) $O(n^3)$
- c) $O(n^2)$
- d) $O(2^n)$

4.5 El siguiente algoritmo calcula las tablas de multiplicar del 1 a n . ¿Cuál es su complejidad asintótica en el peor de los casos?

```

public void tablas(int n)
    for(int i=0; i < n; i++)
        for(int j=0; j < n; j++)
            print (i+"*"+j+"="+i*j);

```

 **Elija la respuesta que considere acertada:**

- a) $O(n)$
- b) $O(2^i)$
- c) $O(i)$
- d) $O(n^2)$

4.6 Consideren el siguiente algoritmo:

```

1 public int misterio(int n, int i){
2     if (i >= n){
3         return n;
4     }
5     return n * misterio(n, i + 1);
6 }

```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

¿Cuántas instrucciones ejecuta el algoritmo en el peor de los casos?

► **Elija la respuesta que considere acertada:**

- a) $T(n) = T(n-1) + C$
- b) $T(n) = T(n-1) + T(n-2) + C$
- c) $T(n) = T(n/2) + C$
- d) $T(n) = T(n+1) + C$

4.7 [Opc] El siguiente algoritmo cuenta del n al 1.

```
public void imprimir(int n) {
    if (n == 1) println(1);
    else { println(n);
          imprimir(n-1);    }}
```

► **4.7.1** ¿Cuál es el número de pasos que ejecuta para el peor de los casos?

$T(n) = \underline{\hspace{2cm}}$

► **4.7.2** ¿Cuál es la complejidad asintótica en el peor de los casos?

$O(\underline{\hspace{2cm}})$

4.8 ¿Cuál de las siguientes afirmaciones es correcta con respecto a func3(n)?

```
1 void func3(int n){
2     if(n < 1) return;
3     else{
4         System.out.println(n);
5         func3(n-1);
6     }
7 }
```

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

► **Elija la respuesta que considere acertada:**

- a) Esta ejecuta $T(n) = c + T(n - 1)$ pasos, que es $O(n)$.
- b) Esta ejecuta $T(n) = n + T(n - 1)$ pasos, que es $O(n!)$.
- c) Esta ejecuta $T(n) = cn + T(n - 1)$ pasos, que es $O(n!)$.
- d) Esta ejecuta $T(n) = c + 2 \cdot T(n - 1)$ pasos, que es $O(2^n)$.

4.9 ¿Cuál de las siguientes afirmaciones es correcta con respecto a la función *func1(n,m)*?

```
void func1(int n, int m){
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n + m; j++){
            print(j);
        }
    }
}
```

► **Elija la respuesta que considere acertada:**

- a) Ejecuta menos de $n+m$ pasos.
- b) Ejecuta exactamente $n+m$ pasos.
- c) Ejecuta $n \times m$ pasos.
- d) Ejecuta más de $n \times m$ pasos.

4.10 [Opc] ¿Cuál de las siguientes afirmaciones es correcta con respecto a la función *func2(n)*?

```
void func2(int n){
    for(int i = 2; i * i <= n; i++){
        for(int k = 2; k * k <= n; k++){
            print(j);
        }
    }
}
```




Elija la respuesta que considere acertada:

- a) Ejecuta más de pasos.
- b) Ejecuta más de pasos.
- c) Ejecuta menos de $n \cdot \log n$ pasos.
- d) Ejecuta exactamente pasos

4.11 ¿Cuál de las siguientes afirmaciones es correcta con respecto a $func3(n)$?

```
int func3(int n){
    if(n == 1 || n == 2){
        return n;
    }
    int ni = func3(n - 1);
    int nj = func3(n - 2);
    int suma = ni + nj;
    return suma;
}
```



Elija la respuesta que considere acertada:

- a) Ejecuta $T(n) = T(n-1) + c$ pasos.
- b) Ejecuta $T(n) = T(n-1) + cn$ pasos.
- c) Ejecuta $T(n) = T(n-1) + T(n-2) + c$ pasos.
- d) Ejecuta $T(n) = T(n/2) + c$ pasos.

4.12

Sea $f(n, m) = n \times \log(n) + m^2$ y $g(n, m) = n + m$. Calcule $O(f(n, m) \times g(n, m))$. Ten en cuenta que si la regla del producto fuera válida para el producto de variables, entonces $O(n \times n)$ sería $O(n)$, lo cual no es cierto. Ten en cuenta que no se sabe cuál es más grande entre n y m .



Elija la respuesta que considere acertada:

- a) $O(n \times \log(n) + m^2)$
- b) $O(m \times n \times \log(n) + n \times m^2 + n^2 \times \log(n) + m^3)$

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

- c) $O(m^3)$
d) $O(n^3 + m^3)$

4.13 [Opc] Considere el siguiente código escrito en Java. Encuentre la ecuación de recurrencia que mejor representa la complejidad asintótica en el peor de los casos.

```
int f(int n){
    if(n <= 0){
        return 1;
    }
    int a = f(n / 2);
    int b = f(n / 2);
    int res = 0;
    for(int i = 0; i < n; i++){
        res += (a*b);
    }
    return res;
}
```

 **Elija la respuesta que considere acertada:**

- a) $T(n)=2T(n-1)+n$
b) $T(n) = 2T(n/2) + n^2$
c) $T(n) = 2T(n/2) + n$
d) $T(n)=2T(n-1)+(n-1)$

4.14 Sea $f(n, m) = n^2 + n \times \log(\log(m))$ y $g(n, m) = n^3 + m \times \sqrt{m}$. Calcule $O(f(n, m) + g(n, m))$. Ten en cuenta que no se sabe cuál es más grande entre n y m .

 **Elija la respuesta que considere acertada:**

ESTRUCTURA DE DATOS 1
Código ST0245

- a)** $O(n^3 + n(\log(\log(m))) + m \times \sqrt{m})$
- b)** $O(n^3)$
- c)** $O(m \times \sqrt{(m)} + n^3)$
- d)** $O(m \times \sqrt{m})$

5. [Opcional] Lecturas Recomendadas

5 [Opc] Lecturas recomendadas



Vean *Guía* en **numeral 3.5 y 4.20**



Exportar y entregar informe de laboratorio en **PDF**, en **español o Inglés**



Si hacen el **informe en español**, usen la **plantilla en español**



No apliquen **Normas Icontec** para esto

Si hacen **el informe en inglés**, usen a **plantilla en inglés**



"El ejercicio de una profesión requiere la adquisición de competencias que se sustentan en procesos comunicativos. Así cuando se entrevista a un ingeniero recién egresado para un empleo, una buena parte de sus posibilidades radica en su capacidad de comunicación; pero se ha observado que esta es una de sus principales debilidades..." *Tomado de <http://bit.ly/2gJKzJD>*



Vean *Guía* en **numerales 3.5 y 4.20**



Lean a **"R.C.T Lee et al., Introducción al análisis y diseño de algoritmos, Capítulo 2. 2005"** y sumen puntos adicionales, así:



Hagan un mapa conceptual con los principales elementos teóricos.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473



Nota: Si desean una lectura adicional en idioma español, consideren la siguiente: **“John Hopcroft et al., Fundamentos de Algoritmia. Capítulo 3: Notación Asintótica. Páginas 11– 98. 1983”** que pueden encontrarla en biblioteca

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473



6. [Opcional] Trabajo en Equipo y Progreso Gradual

6 [Opc] Trabajo en Equipo y Progreso Gradual



Vean Guía en numeral 3.5 y 4.20



Exportar y entregar informe de laboratorio en **PDF**, en **español o Inglés**



Si hacen el **informe en español**, usen la **plantilla en español**



No apliquen **Normas Icontec** para esto

Si hacen **el informe en inglés**, usen a **plantilla en inglés**



El trabajo en equipo es una exigencia del mercado. "Mientras algunos medios retratan la programación como un trabajo solitario, pero requiere mucha comunicación y trabajo grupal. Si trabajas para una compañía, serás parte de un equipo de desarrollo y esperarán que te comuniques bien con otras personas" Tomado de <http://bit.ly/1B6hUDp>



Vean Guía en numerales 3.6, 4.21, 4.22, 4.23

6.1



Entreguen copia de todas las actas de reunión usando el *tablero Kanban*, con fecha, hora e integrantes que participaron

6.2



Entreguen el reporte de *git* con los cambios en el código y quién hizo cada cambio, con fecha, hora e integrantes que participaron

6.3



Entreguen el reporte de cambios del informe de laboratorio que se genera *Google docs* o herramientas similares



Nota: Estas respuestas también deben incluirlas en el informe PDF

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

7. [Opcional]

Laboratorio en Inglés con plantilla en Inglés

7 [Opc] Laboratorio en inglés



Vean Guía en numeral 3.5 y 4.20



Exportar y entregar informe de laboratorio en **PDF**, en **español o Inglés**



Si hacen el **informe en español**, usen la **plantilla en español**



No apliquen **Normas Icontec** para esto

Si hacen **el informe en inglés**, usen a **plantilla en inglés**



El inglés es un idioma importante en la Ingeniería de Sistemas porque la mayoría de los avances en tecnología se publican en este idioma y la traducción, usualmente se demora un tiempo.

Adicionalmente, dominar el inglés permite conseguir trabajos en el exterior que son muy bien remunerados. Tomado de goo.gl/4s3LmZ

► **Entreguen el código y el informe en inglés.**

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Resumen de ejercicios a resolver

1.1 Extiendan el código existente para medir los tiempos de *Insertion Sort* y *Merge sort*

2.1 Resuelvan mínimo 5 ejercicios del nivel *Array 2* de la página *CodingBat*:
<http://codingbat.com/java/Array-2>

2.2 Resuelvan mínimo 5 ejercicios del nivel *Array 3* de la página *CodingBat*:
<http://codingbat.com/java/Array-3>

3.1 De acuerdo a lo realizado en el numeral 1.1, construya una tabla donde muestre, para cada algoritmo, cuánto se demora para cada uno de los tamaños del problema.

3.2 Grafiquen los tiempos que tomó en cada algoritmo para los diferentes tamaños del problema.

3.3 ¿Qué tan eficiente es *merge sort* con respecto a *insertion sort* para arreglos grandes?
¿Es apropiado usar *insertion sort* para una base de datos con millones de elementos?
¿Por qué?

3.4 Expliquen con sus propias palabras cómo funciona su solución del ejercicio de Array 3 de Coding Bat llamado *maxSpan*

3.5 Calculen la complejidad de los ejercicios en línea, numerales 2.1 y 2.2, y agréguelas al informe PDF

3.6 Expliquen con sus palabras las variables (qué es 'n', qué es 'm', etc.) del cálculo de complejidad del numeral anterior

4. Simulacro de Parcial

5. [Ejercicio Opcional] Lectura recomendada

6. [Ejercicio Opcional] Trabajo en Equipo y Progreso Gradual

7. [Ejercicio Opcional] Utilicen la plantilla dispuesta en este idioma para el laboratorio.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

Ayudas para resolver los Ejercicios

Ayudas para el Ejercicio 1.....	<u>Pág. 29</u>
Ayudas para el Ejercicio 3.1 y 3.2	<u>Pág. 29</u>
Ayudas para el Ejercicio 3.7.....	<u>Pág. 29</u>
Ayudas para el Ejercicio 4.....	<u>Pág. 30</u>
Ayudas para el Ejercicio 5.....	<u>Pág. 30</u>
Ayudas para el Ejercicio 6.1.....	<u>Pág. 30</u>
Ayudas para el Ejercicio 6.2.....	<u>Pág. 30</u>
Ayudas para el Ejercicio 6.3.....	<u>Pág. 30</u>



Ayudas para el Ejercicio 1



Pista 1: Si deciden hacer la documentación, consulten la *Guía en numeral 4.1 y 4.6*



Ayudas para el Ejercicio 3.1 y 3.2



Pista 1: Vean *Guía en numeral 4.6* para “Cómo usar la escala logarítmica en Microsoft Excel”.



Pista 2: Si todos los tiempos de un algoritmo dan más de 5 minutos, realice otra tabla, para ese algoritmo, tomando tiempos para arreglos de tamaño 1000, 10000 y 100000.



Pista 3: Vean *Guía en numeral 4.4* para “Cómo aumentar el tamaño del heap y del stack en Java”.



Pista 4: Vean *Guía en numeral 4.5* para “Cómo visualizar el montículo (heap) y el stack, y el consumo total de memoria de Java”.



Ayudas para el Ejercicio 3.7



Pista 1: Vean *Guía en numeral 4.11* para “Cómo escribir la complejidad de un ejercicio en línea”



Pista 2: Vean *Guía en numeral 4.19* para “Ejemplos para calcular la complejidad de un ejercicio de CodingBat”

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473



Ayudas para el Ejercicio 4



Vean la *Guía* en el numeral 4.18

Ayudas para el Ejercicio 5



Pista 1: Para que hagan el mapa conceptual se recomiendan herramientas como las que encuentran en <https://cacoo.com/> o <https://www.mindmup.com/#m:new-a-1437527273469>



Ayudas para el Ejercicio 6.1



Pista 1: Vean Guía en numeral 4.21



Ayudas para el Ejercicio 6.2



Pista 1: Vean Guía en numeral 4.23



Ayudas para el Ejercicio 6.3



Pista 1: Vean Guía en numeral 4.22

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

¿Alguna inquietud?

CONTACTO

Docente Mauricio Toro Bermúdez

Teléfono: (+57) (4) 261 95 00 Ext. 9473

Correo: mtorobe@eafit.edu.co

Oficina: 19- 627

Agenden una cita dando clic en la pestaña

-*Semana*- de <http://bit.ly/2gzVg10>